

Congruence from the Operator’s Point of View: Compositionality Requirements on Process Semantics

Maciej Gazda & Wan Fokkink

Vrije Universiteit
Department of Computer Science
De Boelelaan 1081a, 1081 HV Amsterdam, Netherlands
m.w.gazda@student.vu.nl, wanf@cs.vu.nl

One of the basic sanity properties of a behavioural semantics is that it constitutes a congruence with respect to standard process operators. This issue has been traditionally addressed by the development of rule formats for transition system specifications that define process algebras. In this paper we suggest a novel, orthogonal approach. Namely, we focus on a number of process operators, and for each of them attempt to find the widest possible class of congruences. To this end, we impose restrictions on sublanguages of Hennessy-Milner logic, so that a semantics whose modal characterization satisfies a given criterion is guaranteed to be a congruence with respect to the operator in question. We investigate action prefix, alternative composition, two restriction operators, and parallel composition.

1 Introduction

Congruence is one of the most important properties of a behavioural semantics. The reason is that the fundamental issue in process algebra - providing sound and complete axiomatisations for collections of process operators - requires that these operators are compositional. Only then we can use equational logic principles and provide sound axioms.

There is a large amount of research to find ways of ensuring the congruence property. The basic methodology is to impose restrictions on operator definitions; there is a notion of a rule format for transition system specifications which provide operational semantics for process algebras. If a process operator is defined with rules that fit within a format, then the semantics in question is a congruence with respect to this operator. Examples include the panth format for bisimulation semantics [8] and formats designed specifically for several decorated trace semantics [4]. The focus here is on *semantics*; rule formats are most often defined with one particular process semantics in mind. Interestingly, in [4], the modal characterization of a process semantics is taken as starting point to derive the syntactic constraints of the congruence format for this semantics. A modal characterization of a semantics is a sublanguage of Hennessy-Milner logic such that two processes are semantically equivalent if and only if they satisfy exactly the same formulas in the modal characterization of the semantics. For almost all process semantics in van Glabbeek’s spectrum [6] there is a corresponding modal characterization.

In this paper, we attempt to look at the compositionality issue from an *operator’s* point of view. For a number of basic process operators, we determine conditions that a process semantics should satisfy in order to be congruence with respect to such an operator. To be more precise, given a process operator, we develop syntactic constraints on modal characterizations; if the modal characterization of a process semantics satisfies these constraints, then the process operator is guaranteed to be compositional with respect to this semantics. So instead of going from a process semantics to a class of transition system specifications for which that semantics is a congruence, we go from the transition rules of a process operator to a class of process semantics for which this operator is compositional. This approach gives us

an orthogonal view on compositionality, and provides further insight into connections between process algebra and modal logic.

2 Preliminaries

We work in the usual setting of labelled transition systems (LTSs), which consist of a set S of states p (also called processes), a set Act of actions a , and a set of transitions $p \xrightarrow{a} p'$.

2.1 Hennessy-Milner logic

Hennessy-Milner logic (*HML*) [7] is a modal logic for specifying properties of states in an LTS. There exist different versions of *HML* [7, 6, 3]. The choice of syntax is important here, even if two logics have the same expressivity; compositionality requirements established for some version of *HML* (e.g. with diamond, conjunction and negation only) may become insufficient when we add other operators (e.g. box), because these extra operators may require syntactic requirements of their own. Our point of departure is the infinitary *HML* variant without box and disjunction. The *HML* syntax is therefore as follows:

$$\varphi ::= \top \mid \bigwedge_{i \in I} \varphi_i \mid \langle a \rangle \varphi \mid \neg \varphi$$

where I is an arbitrary index set, and a ranges over the set Act of actions. Furthermore, we use F as an abbreviation for $\neg \top$. We introduce some additional notations, based on the standard notion of context. A context, notation $C[\]$, is a *HML* formula with one occurrence of \square . A *multicontext* $C[\]_{i \in I}$ is a *HML* formula containing one or more \square symbols, indexed by the elements from I . For a (multi)context $C[\varphi_i]_{i \in I}$, a formula is obtained by replacing the \square_i symbols with formulas φ_i . Finally, we introduce an *n-level context*, which means that the context symbol has n diamond operators above it. It is defined inductively as follows:

- \square is a **0-level** context;
- if $C_n[\]$ is an n -level context, then $\neg C_n[\]$ and $C_n[\] \wedge \bigwedge_{i \in I} \varphi_i$ are n -level contexts;
- if $C_n[\]$ is an n -level context, then $\langle a \rangle C_n[\]$ is an $(n+1)$ -level context.

An example of a **0-level** context is $\langle a \rangle \langle b \rangle \top \wedge \neg \square$, while $\langle a \rangle (\langle a \rangle \langle b \rangle \top \wedge \square)$ is a **1-level** context.

A sublanguage \mathcal{O} of *HML* gives rise to a process equivalence by identifying those processes which satisfy exactly the same formulas from \mathcal{O} :

$$p \sim_{\mathcal{O}} q \stackrel{\text{def}}{\iff} \forall \varphi \in \mathcal{O} : (p \models \varphi \iff q \models \varphi).$$

We call \mathcal{O} a modal characterization of $\sim_{\mathcal{O}}$. Below, examples of modal characterizations of standard process equivalences from the literature are given (see [6]):

- *trace observations*:
 $\mathcal{O}_T \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_T)$
- *completed trace observations*:
 $\mathcal{O}_{CT} \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_{CT}) \mid \bigwedge_{a \in Act} \neg \langle a \rangle \top$
- *failures observations*:
 $\mathcal{O}_F \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_F) \mid \bigwedge_{i \in I} \neg \langle a_i \rangle \top$
- *readiness observations*:
 $\mathcal{O}_R \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_R) \mid \bigwedge_{i \in I} \neg \langle a_i \rangle \top \wedge \bigwedge_{j \in J} \langle b_j \rangle \top$

- *failure trace observations:*

$$\mathcal{O}_{FT} \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_{FT}) \mid \bigwedge_{i \in I} \neg \langle a_i \rangle \top \wedge \varphi' \ (\varphi' \in \mathcal{O}_{FT})$$

- *ready trace observations:*

$$\mathcal{O}_{RT} \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_{RT}) \mid \bigwedge_{i \in I} \neg \langle a_i \rangle \top \wedge \bigwedge_{j \in J} \langle b_j \rangle \top \wedge \varphi' \ (\varphi' \in \mathcal{O}_{RT})$$

- *simulation observations:*

$$\mathcal{O}_{1S} \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_{1S}) \mid \bigwedge_{i \in I} \varphi_i \ (\varphi_i \in \mathcal{O}_{1S})$$

- *ready simulation observations:*

$$\mathcal{O}_{RS} \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_{RS}) \mid \neg \langle a \rangle \top \mid \bigwedge_{i \in I} \varphi_i \ (\varphi_i \in \mathcal{O}_{RS})$$

- *n-nested simulation observations for $n \geq 2$:*

$$\mathcal{O}_{nS} \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_{nS}) \mid \bigwedge_{i \in I} \varphi_i \ (\varphi_i \in \mathcal{O}_{nS}) \mid \neg \varphi' \ (\varphi' \in \mathcal{O}_{(n-1)S})$$

- *bisimulation observations:*

$$\mathcal{O}_B \ \varphi ::= \top \mid \langle a \rangle \varphi' \ (\varphi' \in \mathcal{O}_B) \mid \bigwedge_{i \in I} \varphi_i \ (\varphi_i \in \mathcal{O}_B) \mid \neg \varphi' \ (\varphi' \in \mathcal{O}_B)$$

We write $\varphi \equiv \varphi'$ if $p \models \varphi \Leftrightarrow p \models \varphi'$ for any process p in any LTS. Given an $\mathcal{O} \subseteq HML$, we write \mathcal{O}^\equiv for the set of HML formulas φ for which there exists a $\varphi' \in \mathcal{O}$ with $\varphi \equiv \varphi'$.

2.2 BCCSP

Any LTS isomorphic with a finite tree can be described with the following process algebra BCCSP, consisting of three operators:

- a nullary process $\mathbf{0}$ which does not have any behaviour;
- action prefix $a.()$ for $a \in Act$: a unary operator which represents execution of a single action followed by the process given as the argument, defined by the transition rule

$$\frac{}{ax \xrightarrow{a} x}$$

- alternative composition $(+)$, a nondeterministic choice between two processes, defined by the transition rules

$$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \quad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$$

In this paper we focus on several process operators from the literature, and try to establish which syntactic properties a modal language $\mathcal{O} \subseteq HML$ should satisfy to guarantee that the induced equivalence is a congruence with respect to the given operator. That is, given a process operator f , we will search for a syntactic condition C such that if \mathcal{O} satisfies C , then f is compositional with respect to $\sim_{\mathcal{O}}$.

3 Basic operators

3.1 Alternative composition

We start with alternative composition, which expresses a nondeterministic choice between two processes. We want to find a general property of a modal language that would guarantee congruence of the induced

equivalence with respect to alternative composition. Our first observation is that the behaviour of an alternative composition $p_1 + p_2$ after performing the first step is completely determined by the behaviour of one of the components. For example, $p_1 + p_2 \models \langle a \rangle \varphi$ if and only if either $p_1 \models \langle a \rangle \varphi$ or $p_2 \models \langle a \rangle \varphi$. The only potential problem can occur when there is a formula with a conjunction at level 0 (i.e., not in the scope of an action prefix). For instance, consider $\mathcal{O} = \{\langle a \rangle \top \wedge \langle b \rangle \top\}$. We have $a\mathbf{0} \sim_{\mathcal{O}} \mathbf{0}$ and $b\mathbf{0} \sim_{\mathcal{O}} \mathbf{0}$, but $a\mathbf{0} + b\mathbf{0} \not\sim_{\mathcal{O}} \mathbf{0} + \mathbf{0}$. As it turns out, it suffices to simply close the language on sub-conjunctions at level 0.

Theorem 1 Let $\mathcal{O} \subseteq \text{HML}$. If for any 0-level context $C_0[]$ and $\varphi_i \in \text{HML}$ for $i \in I$,

$$(\text{AC}) \quad C_0[\bigwedge_{i \in I} \varphi_i] \in \mathcal{O} \text{ implies that } \forall_{i \in I} : (\varphi_i \in \mathcal{O}^{\equiv}),$$

then $\sim_{\mathcal{O}}$ is a congruence with respect to alternative composition $(+)$.

Proof: Assume a modal language \mathcal{O} with the AC property. Let $p_1 \sim_{\mathcal{O}} q_1$ and $p_2 \sim_{\mathcal{O}} q_2$. We show that for any $\varphi \in \mathcal{O}$:

$$p_1 + p_2 \models \varphi \Rightarrow q_1 + q_2 \models \varphi$$

(the converse implication " \Leftarrow " is symmetric). We apply induction on the structure of φ . The base case (T) is trivial. We proceed with the inductive step. Assume that $p_1 + p_2 \models \varphi$. We have to consider the following cases:

- $\varphi = \langle a \rangle \psi$: then either $p_1 \models \langle a \rangle \psi$ or $p_2 \models \langle a \rangle \psi$. From the equivalence of components we have either $q_1 \models \langle a \rangle \psi$ or $q_2 \models \langle a \rangle \psi$, which yields $q_1 + q_2 \models \langle a \rangle \psi$.
- $\varphi = \bigwedge_{i \in I} \varphi_i$: we have $p_1 + p_2 \models \bigwedge_{i \in I} \varphi_i \Leftrightarrow \forall_{i \in I} : p_1 + p_2 \models \varphi_i \Leftrightarrow \forall_{i \in I} : q_1 + q_2 \models \varphi_i$ (AC + inductive hypothesis) $\Leftrightarrow q_1 + q_2 \models \bigwedge_{i \in I} \varphi_i$.
- $\varphi = \neg \psi$: let ψ' be the outermost subformula of φ which does not begin with a " \neg " symbol (so $\varphi = (\neg)^n \psi'$). Then φ is logically equivalent to either ψ' or $\neg \psi'$. The case $\psi' = \top$ is trivial. Also, the case where $\varphi \equiv \psi'$ can be handled analogously as the first two cases. We thus have to consider two possibilities:
 - $\varphi \equiv \neg \langle a \rangle \varphi'$: we have $p_1 + p_2 \models \neg \langle a \rangle \varphi' \Leftrightarrow p_1 \models \neg \langle a \rangle \varphi' \wedge p_2 \models \neg \langle a \rangle \varphi' \Leftrightarrow q_1 \models \neg \langle a \rangle \varphi' \wedge q_2 \models \neg \langle a \rangle \varphi'$ (equivalence of components) $\Leftrightarrow q_1 + q_2 \models \neg \langle a \rangle \varphi'$.
 - $\varphi \equiv \neg \bigwedge_{i \in I} \varphi_i$: we have $p_1 + p_2 \models \neg \bigwedge_{i \in I} \varphi_i \Leftrightarrow \exists_{i \in I} : p_1 + p_2 \models \neg \varphi_i \Leftrightarrow \exists_{i \in I} : q_1 + q_2 \models \neg \varphi_i$ (AC + inductive hypothesis) $\Leftrightarrow q_1 + q_2 \models \neg \bigwedge_{i \in I} \varphi_i$.

□

For example, consider $\mathcal{O} = \{\langle a \rangle (\langle a \rangle \top \wedge \langle b \rangle \top) \wedge \neg \langle b \rangle \top, \langle a \rangle (\langle a \rangle \top \wedge \langle b \rangle \top), \neg \langle b \rangle \top\}$. The language \mathcal{O} satisfies the AC requirement, and so the corresponding equivalence $\sim_{\mathcal{O}}$ is a congruence with respect to $+$.

Almost all modal characterizations of standard process semantics from Section 2.1 fulfill AC. The only exception is the modal characterization of completed trace equivalence, although we can provide an alternative characterization that meets the AC requirement:

$$\mathcal{O}_{CT}^* \quad \varphi ::= \varphi' \ (\varphi' \in \mathcal{O}_{CT}) \mid \neg \langle a \rangle \top$$

The characterization \mathcal{O}_{CT}^* is the same as \mathcal{O}_{CT} , except that it includes formulas $\neg \langle a \rangle \top$. Clearly this does not change the corresponding semantics.

3.2 Action prefix

In the case of action prefix, it is easy to obtain a sufficient congruence requirement; the crucial observation is that $a.p \models \langle a \rangle \phi$ if and only if $p \models \phi$, so we need to make sure that for each formula $C_0[\langle a \rangle \phi] \in \mathcal{O}$, the subformula ϕ also belongs to the language \mathcal{O} . If this is not the case, an equivalence might not be a congruence. For instance, if $\mathcal{O} = \{\langle a \rangle \langle a \rangle \top\}$, then $a.0 \sim_{\mathcal{O}} 0$, but $a.a.0 \not\sim_{\mathcal{O}} a.0$.

Theorem 2 Let $\mathcal{O} \subseteq HML$ and fix $a \in Act$. If for any 0-level context $C_0[]$ and $\phi \in HML$,

$$(AP) \quad C_0[\langle a \rangle \phi] \in \mathcal{O} \text{ implies that } \phi \in \mathcal{O}^{\equiv},$$

then $\sim_{\mathcal{O}}$ is a congruence with respect to the action prefix operator $a.()$.

Proof: Let $p \sim_{\mathcal{O}} q$. We need to show that for any $\phi \in \mathcal{O}$, $a.p \models \phi \Leftrightarrow a.q \models \phi$.

Take any $\phi \in \mathcal{O}$. Let $\phi = C[\langle a_i \rangle \phi_i]_{i \in I}$ such that the multicontext $C[]_{i \in I}$ does not contain any action prefix symbols. That is, the $\langle a_i \rangle \phi_i$ for $i \in I$ are all action prefix subformulas of ϕ that appear at level zero. Since $C[]_{i \in I}$ is built from only \top , conjunction and negation, whether a process satisfies ϕ is completely determined by the satisfiability of $\langle a_i \rangle \phi_i$ for $i \in I$ by this process. In other words, if $\forall_{i \in I} : (p_1 \models \langle a_i \rangle \phi_i \Leftrightarrow p_2 \models \langle a_i \rangle \phi_i)$, then $p_1 \models \phi \Leftrightarrow p_2 \models \phi$. Coming back to our setting with $a.p$ and $a.q$, take an arbitrary $i \in I$. We have:

$$\begin{aligned} a.p &\models \langle a_i \rangle \phi_i \\ &\Leftrightarrow (a_i = a) \wedge p \models \phi_i \\ &\Leftrightarrow (a_i = a) \wedge q \models \phi_i \text{ (AP + } p \sim_{\mathcal{O}} q) \\ &\Leftrightarrow a.q \models \langle a_i \rangle \phi_i. \end{aligned}$$

The choice of i was arbitrary, hence the earlier remark yields: $a.p \models \phi \Leftrightarrow a.q \models \phi$. □

The AP condition is satisfied by all modal characterizations from Section 2.1.

4 Restriction operators: projection and encapsulation

We now consider projection and encapsulation operators. The n th projection of a process p , for $n \geq 0$, mimicks the behaviour of p up to level n :

$$\frac{x \xrightarrow{a} x'}{\pi_{n+1}(x) \xrightarrow{a} \pi_n(x')}$$

Applying encapsulation with parameter $B \subseteq Act$ removes all transitions whose labels are in B from the process:

$$\frac{x \xrightarrow{a} x' \quad (a \notin B)}{\partial_B(x) \xrightarrow{a} \partial_B(x')}$$

More generally, we consider unary restriction operators f such that given a process p , the process $f(p)$ can be viewed as a subgraph of p . Below we will give a precise description of which restriction operators are covered. For the projection operator π_n as well as for the encapsulation operator ∂_B , given any HML formula we can deduce in advance which of its subformulas $\langle a \rangle \phi$ will always yield false, regardless of the process $\pi_n(p)$ or $\partial_B(p)$ for which the HML formula is evaluated. In case of a process $\pi_n(p)$, any subformula $\langle a \rangle \phi$ that appears at level n can be replaced by F . And in case of a process $\partial_B(p)$, any subformula $\langle b \rangle \phi$ with $b \in B$ can be replaced by F .

We cannot reason in this way about any restriction operator. For example, consider the priority operator θ , which assumes a partial order $<$ on the set of actions and allows us to execute an action only if no action with higher priority is executable at the same time:

$$\frac{x \xrightarrow{a} x' \quad \forall b \in \text{Act} (a < b \Rightarrow x \not\xrightarrow{b})}{\theta(x) \xrightarrow{a} \theta(x')}$$

Suppose that $a > b$ and there is a process p of which we only know that it satisfies $\langle b \rangle \top$. This knowledge is not sufficient to determine whether $\theta(p) \models \langle b \rangle \top$.

Let f be a unary operator such as π_n or ∂_B . We would like to define for each formula $\varphi \in \text{HML}$ a corresponding formula $\text{cut}_f(\varphi)$ in which every subformula $\langle a \rangle \varphi'$ which is known in advance to be unsatisfiable when evaluating any process $f(p)$ is replaced by F . Actually this means that either we can replace a larger subformula by \top , or the entire formula becomes F . Namely, we can replace the first innermost negation symbol (closest to the introduced F) and the following subformula by \top ; if the F symbol does not appear within the scope of a negation symbol, then the whole formula yields F . If a language $\mathcal{O} \subseteq \text{HML}$ is closed under cut_f , then it induces a congruence with respect to f . The whole idea is made formal below.

Lemma 1 Let f be a unary process operator. Suppose there exists a function $\text{cut}_f : \text{HML} \rightarrow \text{HML}$ such that for any process p and $\varphi \in \text{HML}$,

$$(\text{CUT}) \quad f(p) \models \varphi \Leftrightarrow p \models \text{cut}_f(\varphi)$$

Then for any language \mathcal{O} satisfying

$$\varphi \in \mathcal{O} \Rightarrow (\text{cut}_f(\varphi) \in \mathcal{O}^\equiv \vee \text{cut}_f(\varphi) \equiv F)$$

the corresponding equivalence $\sim_{\mathcal{O}}$ is a congruence with respect to f .

Proof: Suppose $\mathcal{O} \subseteq \text{HML}$ and $p \sim_{\mathcal{O}} q$. We have $f(p) \models \varphi \Leftrightarrow p \models \text{cut}_f(\varphi)$ (CUT) $\Leftrightarrow q \models \text{cut}_f(\varphi)$ (either because $\text{cut}_f(\varphi) \in \mathcal{O}^\equiv$ and $p \sim_{\mathcal{O}} q$, or because $\text{cut}_f(\varphi) \equiv F \Leftrightarrow f(q) \models \varphi$ (CUT)). \square

The next lemma gives an explicit condition for a modal language to induce a congruence in case cut_f formulas are obtained from the original ones by turning certain subformulas $\langle a \rangle \varphi$ into F .

Lemma 2 Assume f and cut_f are as in Lem. 1, and satisfy CUT. Suppose that for each $\varphi \in \text{HML}$ there exists a multicontext $C[\]_{i \in I}$ such that $\varphi = C[\langle a_i \rangle \varphi_i]_{i \in I}$ and $\text{cut}_f(\varphi) \equiv C[F]_{i \in I}$. Then for each language $\mathcal{O} \subseteq \text{HML}$ that satisfies for any context $C'[\]$ and $\varphi \in \text{HML}$,

$$(\text{RES}) \quad C'[\neg \varphi] \in \mathcal{O} \text{ implies } C'[\top] \in \mathcal{O}^\equiv,$$

the corresponding equivalence $\sim_{\mathcal{O}}$ is a congruence with respect to f .

Proof: By Lem. 1 it suffices to prove that for all $\varphi \in \mathcal{O}$ either $\text{cut}_f(\varphi) \in \mathcal{O}$ or $\text{cut}_f(\varphi) \equiv F$. Take any $\varphi \in \mathcal{O}$ such that $\text{cut}_f(\varphi) \not\equiv F$. By assumption, $\text{cut}_f(\varphi) \equiv C[F]_{i \in I}$ for some multicontext $C[\]_{i \in I}$. Since $\text{cut}_f(\varphi) \not\equiv F$, clearly each occurrence of F in this formula must be within the scope of a negation symbol. Hence $\text{cut}_f(\varphi) \equiv C'[\neg D^i[F]]_{i \in I}$, where we can choose contexts $D^i[\]$ for $i \in I$ such that in each $D^i[\]$, $\]$ is not within the scope of a negation. Then $C'[\neg D^i[F]]_{i \in I} \equiv C'[\top]_{i \in I}$. Since \mathcal{O} satisfies RES, $C'[\top]_{i \in I} \in \mathcal{O}^\equiv$. Hence $\text{cut}_f(\varphi) \in \mathcal{O}^\equiv$. \square

We have provided a compositionality framework for a general class of restriction operators. What remains is to provide cut_f functions for the projection and encapsulation operators.

Lemma 3 The functions cut_f defined below are proper cutting functions (i.e., they satisfy condition CUT of Lem. 1).

a) For the projection operators π_n with $n \geq 0$:

$$\begin{aligned} cut_n(\top) &= \top & cut_n(\bigwedge_{i \in I} \varphi_i) &= \bigwedge_{i \in I} cut_n(\varphi_i) & cut_n(\neg \varphi) &= \neg cut_n(\varphi) \\ cut_0(\langle a \rangle \varphi) &= \text{F} & cut_{n+1}(\langle a \rangle \varphi) &= \langle a \rangle cut_n(\varphi) \end{aligned}$$

b) For the encapsulation operators ∂_B with $B \subseteq \text{Act}$:

$$\begin{aligned} cut_B(\top) &= \top & cut_B(\bigwedge_{i \in I} \varphi_i) &= \bigwedge_{i \in I} cut_B(\varphi_i) & cut_B(\neg \varphi) &= \neg cut_B(\varphi) \\ cut_B(\langle a \rangle \varphi) &= \text{F} \text{ if } a \in B & cut_B(\langle a \rangle \varphi) &= \langle a \rangle cut_B(\varphi) \text{ if } a \notin B \end{aligned}$$

Proof: a) We prove CUT by induction on the structure of φ .

- $\varphi = \top$:

$$\pi_n(p) \models \top \text{ and } p \models cut_n(\top) = \top.$$

- $\varphi = \langle a \rangle \psi$:

We distinguish the cases $n = 0$ and $n > 0$. Clearly $\pi_0(p) \not\models \langle a \rangle \psi$ and $p \not\models cut_0(\langle a \rangle \psi) = \text{F}$.

If $n > 0$, then $\pi_n(p) \models \langle a \rangle \psi \Leftrightarrow \exists p' : p \xrightarrow{a} p' \wedge \pi_{n-1}(p') \models \psi$ (transition rule for π_n) $\Leftrightarrow \exists p' : p \xrightarrow{a} p' \wedge p' \models cut_{n-1}(\psi)$ (structural induction) $\Leftrightarrow p \models \langle a \rangle cut_{n-1}(\psi) \Leftrightarrow p \models cut_n(\langle a \rangle \psi)$ (definition of cut_n).

- $\varphi = \bigwedge_{i \in I} \psi_i$:

$$\pi_n(p) \models \bigwedge_{i \in I} \psi_i \Leftrightarrow \forall_{i \in I} : \pi_n(p) \models \psi_i \Leftrightarrow \forall_{i \in I} : p \models cut_n(\psi_i) \text{ (structural induction)} \Leftrightarrow p \models cut_n(\bigwedge_{i \in I} \psi_i) \text{ (definition of } cut_n).$$

- $\varphi = \neg \psi$:

$$\pi_n(p) \models \neg \psi \Leftrightarrow \pi_n(p) \not\models \psi \Leftrightarrow p \not\models cut_n(\psi) \text{ (structural induction)} \Leftrightarrow p \models \neg cut_n(\psi) \Leftrightarrow p \models cut_n(\neg \psi) \text{ (definition of } cut_n).$$

b) Again we use structural induction on φ .

- $\varphi = \top$:

$$\partial_B(p) \models \top \text{ and } p \models cut_B(\top) = \top.$$

- $\varphi = \langle a \rangle \psi$:

Suppose first that $a \in B$. Then $\partial_B(p) \not\models \langle a \rangle \psi$ (transition rule for ∂_B) and $p \not\models cut_B(\langle a \rangle \psi) = \text{F}$ (definition of cut_B).

Suppose now that $a \notin B$. Then $\partial_B(p) \models \langle a \rangle \psi \Leftrightarrow \exists p' : p \xrightarrow{a} p' \wedge \partial_B(p') \models \psi$ (transition rule for ∂_B) $\Leftrightarrow \exists p' : p \xrightarrow{a} p' \wedge p' \models cut_B(\psi)$ (structural induction) $\Leftrightarrow p \models \langle a \rangle cut_B(\psi) \Leftrightarrow p \models cut_B(\langle a \rangle \psi)$ (definition of cut_B).

- $\varphi = \bigwedge_{i \in I} \psi_i$:

$$\partial_B(p) \models \bigwedge_{i \in I} \psi_i \Leftrightarrow \forall_{i \in I} : \partial_B(p) \models \psi_i \Leftrightarrow \forall_{i \in I} : p \models cut_B(\psi_i) \text{ (structural induction)} \Leftrightarrow p \models cut_B(\bigwedge_{i \in I} \psi_i) \text{ (definition of } cut_B).$$

- $\varphi = \neg \psi$:

$$\partial_B(p) \models \neg \psi \Leftrightarrow \partial_B(p) \not\models \psi \Leftrightarrow p \not\models cut_B(\psi) \text{ (structural induction)} \Leftrightarrow p \models \neg cut_B(\psi) \Leftrightarrow p \models cut_B(\neg \psi) \text{ (definition of } cut_B).$$

□

Theorem 3 For any language $\mathcal{O} \subseteq HML$ satisfying RES, the corresponding equivalence $\sim_{\mathcal{O}}$ is a congruence with respect to the projection operators π_n (for $n \geq 0$) and the encapsulation operators ∂_B (for $B \subseteq Act$).

Proof: By Lem. 3, the functions cut_n and cut_B satisfy CUT. Observe that the cut_n and cut_B functions defined in the Lem. 3 only replace certain subformulas $\langle a \rangle \psi$ of the original formula with F. So they meet the requirements of Lem. 2. Congruence is thus an immediate consequence of Lem. 2. □

To demonstrate that the RES requirement is essential, consider the following counterexamples.

- For projection, take $\mathcal{O} = \{\neg \langle a \rangle \neg \langle a \rangle T\}$. We have $aa0 \sim_{\mathcal{O}} 0$, but $\pi_1(aa0) \not\sim_{\mathcal{O}} \pi_1(0)$.
- For encapsulation, take $\mathcal{O} = \{\langle a \rangle \neg \langle b \rangle T\}$. We have $ab0 \sim_{\mathcal{O}} 0$, but $\partial_{\{b\}}(ab0) \not\sim_{\mathcal{O}} \partial_{\{b\}}(0)$.

The RES requirement is satisfied by every characterization from Section 2.1, except for completed trace observations. Completed trace equivalence is a congruence with respect to projection operators, but not encapsulation. Take for instance the completed trace equivalent processes $a(b0 + c0)$ and $ab0 + ac0$. We have $\partial_{\{b\}}(a(b0 + c0)) \sim_{CT} ac0 \not\sim_{CT} a0 + ac0 \sim_{CT} \partial_{\{b\}}(ab0 + ac0)$.

5 Parallel composition (\parallel)

We now consider the parallel composition operator (without communication). That is, $p \parallel q$ behaves as $p \parallel_{\perp} q + q \parallel_{\perp} p$ where the left-merge operator is defined by

$$\frac{x \xrightarrow{a} x'}{x \parallel_{\perp} y \xrightarrow{a} x' \parallel y}$$

Let us restrict for a moment to only trace formulas (meaning that conjunctions are disregarded). The following example shows that the requirement AP and even being closed under substrings is not sufficient (by a substring of w we mean a subsequence consisting of elements appearing *consecutively* in w). Take $\mathcal{O} = \{\langle a \rangle T, \langle b \rangle T, \langle a \rangle \langle b \rangle T, \langle a \rangle \langle b \rangle \langle a \rangle T, \langle b \rangle \langle a \rangle T\}$. This language not only satisfies AP, but is also closed under prefixes and substrings (but not arbitrary subsequences). However, we have $aa0 \sim_{\mathcal{O}} a0$, but $aa0 \parallel b0 \models \langle a \rangle \langle b \rangle \langle a \rangle T$ while $a0 \parallel b0$ does not satisfy this formula.

This example suggests that if a trace σ belongs to \mathcal{O} , then all *subsequences* of σ must belong to the language as well. This is not unexpected; the behaviour of parallel composition consists of all possible interleavings of the component processes, and all of these interleavings should be described in the modal characterization.

It is also necessary to close the language on subconjunctions. Indeed, take $\mathcal{O} = \{\langle a \rangle T \wedge \langle b \rangle T\}$, a language which does not meet this condition. We have $a0 \sim_{\mathcal{O}} b0$, but $a0 \parallel b0 \models \langle a \rangle T \wedge \langle b \rangle T$ while $b0 \parallel b0$ does not satisfy this formula.

In case of general *HML* formulas, we first define a generalization of a subsequence for an arbitrary formula $\varphi \in HML$ by specifying a set of subformulas with possible replacement from a lower level. We thus define $Sub(\varphi)$ as the smallest set of *HML* formulas satisfying:

- $\varphi \in Sub(\varphi)$;
- $\varphi' \in Sub(\varphi) \Rightarrow \{D[\psi] \mid \varphi' = D[C[\psi]]\} \subseteq Sub(\varphi)$.

We now define a tool to infer satisfaction of modal formulas by a parallel composition $p||q$ from the formulas satisfied by the component processes p and q . This is accomplished by the function Par , which given $A \subseteq HML$ and $B \subseteq HML$, returns the collection of formulas that are certainly satisfied by a parallel composition of two processes satisfying A and B respectively. One can view $Par(A, B)$ as parallel composition operator on collections of modal formulas.

Formally, $Par : \mathcal{P}(HML) \times \mathcal{P}(HML) \longrightarrow \mathcal{P}(HML)$ is defined with induction on the structure of formulas.

- $\top \in Par(A, B)$
 - $\langle a \rangle \phi \in Par(A, B) \stackrel{\text{def}}{\iff} (\exists \langle a \rangle \phi_A \in A : \phi \in Par(\phi_A, B)) \vee (\exists \langle a \rangle \phi_B \in B : \phi \in Par(A, \phi_B))$
 - $\bigwedge_{i \in I} \phi_i \in Par(A, B) \stackrel{\text{def}}{\iff} \forall_{i \in I} : \phi_i \in Par(A, B)$
 - $\neg \phi \in Par(A, B) \stackrel{\text{def}}{\iff} \forall C, D \subseteq Sub(\phi) : \phi \in Par(C, D) \implies (\exists \psi_C \in C : \neg \psi_C \in A) \vee (\exists \psi_D \in D : \neg \psi_D \in B)$
- By abuse of notation, we let $A \subseteq HML$ also denote the formula $\bigwedge_{\phi \in A} \phi$.

Lemma 4 Let $\phi \in HML$.

$$p||q \models \phi \iff \exists A, B \subseteq Sub(\phi) : (p \models A \wedge q \models B \wedge \phi \in Par(A, B)).$$

Proof: We use induction on the structure of formulas. The base case ($\phi = \top$) is immediate. We proceed with the inductive step:

" \Rightarrow ": Assume that $p||q \models \phi$. We prove that $\exists A, B \subseteq Sub(\phi) : (p \models A \wedge q \models B \wedge \phi \in Par(A, B))$.

- $\phi = \langle a \rangle \psi$: Without loss of generality, suppose $p||_q \models \langle a \rangle \psi$ (the case $q||_p \models \langle a \rangle \psi$ is symmetric), so $p \xrightarrow{a} p' \wedge p' || q \models \psi$. From the inductive hypothesis we know that there are $A', B' \subseteq Sub(\psi)$ such that $p' \models A' \wedge q \models B' \wedge \psi \in Par(A', B')$. We take $A = \langle a \rangle (\bigwedge_{\phi \in A'} \phi)$ and $B = B'$.
- $\phi = \bigwedge_{i \in I} \phi_i$: By the inductive hypothesis, for each $i \in I$ there are $A_i, B_i \subseteq Sub(\phi_i)$ such that $p \models A_i \wedge q \models B_i \wedge \phi_i \in Par(A_i, B_i)$. We can take $A = \bigcup_{i \in I} A_i$ and $B = \bigcup_{i \in I} B_i$.

- $\phi = \neg \psi$: We have:

$$\begin{aligned} p||q \models \neg \psi &\iff \neg(p||q \models \psi) \\ &\iff \neg(\exists C, D \subseteq Sub(\psi) : (p \models C \wedge q \models D \wedge \psi \in Par(C, D))) \text{ (inductive hypothesis)} \\ &\iff \forall C, D \subseteq Sub(\psi) : \psi \in Par(C, D) \implies (\exists \psi_C \in C : p \not\models \psi_C) \vee (\exists \psi_D \in D : q \not\models \psi_D). \end{aligned}$$

We define:

$$\begin{aligned} A_p(\psi) &= \bigcup_{C, D \subseteq Sub(\psi) : \psi \in Par(C, D)} \{ \neg \psi_C \mid p \not\models \psi_C \wedge \psi_C \in C \} \\ B_q(\psi) &= \bigcup_{C, D \subseteq Sub(\psi) : \psi \in Par(C, D)} \{ \neg \psi_D \mid q \not\models \psi_D \wedge \psi_D \in D \} \end{aligned}$$

These are the A and B we are looking for.

" \Leftarrow ": Suppose that $\exists A, B \subseteq Sub(\phi) : (p \models A, q \models B \wedge \phi \in Par(A, B))$. We prove that $p||q \models \phi$.

- $\phi = \langle a \rangle \psi$: From $\langle a \rangle \psi \in Par(A, B)$ we have $(\exists \langle a \rangle \psi_A \in A : \psi \in Par(\psi_A, B)) \vee (\exists \langle a \rangle \psi_B \in B : \psi \in Par(A, \psi_B))$. Without loss of generality suppose that $(\exists \langle a \rangle \psi_A \in A : \psi \in Par(\psi_A, B))$. Then $p \xrightarrow{a} p' : p' \models \psi_A$. From $\psi \in Par(\psi_A, B)$ and the inductive hypothesis we have $p' || q \models \psi$. Since $p||q \xrightarrow{a} p' || q$, we finally obtain $p||q \models \langle a \rangle \psi$.
- $\phi = \bigwedge_{i \in I} \phi_i$: According to the definition of Par we have $\forall_{i \in I} : \phi_i \in Par(A, B)$. The inductive hypothesis yields $\forall_{i \in I} : p||q \models \phi_i$, and hence $p||q \models \bigwedge_{i \in I} \phi_i$.
- $\phi = \neg \psi$: We have $\forall C, D \subseteq Sub(\psi) : \psi \in Par(C, D) \implies (\exists \psi_C \in C : \neg \psi_C \in A) \vee (\exists \psi_D \in D : \neg \psi_D \in B)$. Suppose, towards a contradiction, that $p||q \not\models \psi$. Then according to the inductive hypothesis there exist C, D such that $p \models C, q \models D$ and $\psi \in Par(C, D)$. But from the earlier remark, we have either $\in C : \neg \psi_C \in A$ or $\psi_D \in D : \neg \psi_D \in B$. This contradicts the fact that $p \models A, C$ and $q \models B, D$.

□

Theorem 4 For any language $\mathcal{O} \subseteq HML$ satisfying

$$(PAR) \quad \varphi \in \mathcal{O} \Rightarrow Sub(\varphi) \subseteq \mathcal{O}^\equiv,$$

$\sim_{\mathcal{O}}$ is a congruence with respect to parallel composition \parallel .

Proof: Suppose $p_1 \sim_{\mathcal{O}} q_1$ and $p_2 \sim_{\mathcal{O}} q_2$. Suppose that $p_1 \parallel p_2 \models \varphi \in \mathcal{O}$. According to Lem. 4, there exist $A, B \subseteq Sub(\varphi)$ such that $p_1 \models A, p_2 \models B$ and $\varphi \in Par(A, B)$. Since $\varphi \in \mathcal{O}$, by condition PAR, $A, B \subseteq \mathcal{O}^\equiv$. Since $p_1 \sim_{\mathcal{O}} q_1$ and $p_2 \sim_{\mathcal{O}} q_2$, it follows that $q_1 \models A$ and $q_2 \models B$. According to Lem. 4 this implies $q_1 \parallel q_2 \models \varphi$.

□

As an example, if we want to define a modal language that would be a congruence with respect to parallel composition, which includes behaviour described by a formula $\langle a \rangle (\neg \langle b \rangle T \wedge \langle c \rangle \langle d \rangle T)$, we should include the following formulas in the characterization (we omit irrelevant formulas like $\langle a \rangle \neg T$): $\langle a \rangle T$, $\langle a \rangle \neg \langle b \rangle T$, $\langle a \rangle \langle c \rangle T$, $\langle a \rangle \langle c \rangle \langle d \rangle T$, $\langle a \rangle \langle d \rangle T$, $\langle a \rangle (\neg \langle b \rangle T \wedge \langle c \rangle T)$, $\langle a \rangle (\neg \langle b \rangle T \wedge \langle d \rangle T)$.

All basic equivalences except for completed trace have modal characterizations that satisfy the condition PAR. We note that parallel composition is compositional with respect to completed trace equivalence.

6 Conclusions and future work

We have presented, for a number of process operators from the literature, general conditions that guarantee congruence of process equivalences defined by means of a modal characterization. To the best of our knowledge it is the first such attempt.

Our conditions are sufficient, but by no means necessary. We believe that it is difficult (if not impossible) to provide a syntactic restriction on a modal language that would characterize the class of congruences for a given operator (strictly speaking, languages that induce congruences). We aimed at clear and comprehensible rather than slightly relaxed but more complicated conditions.

As the next step, we would like to investigate other process operators (e.g. sequential composition, renaming, merge with communication), consider the setting of weak semantics and different modal languages. In the last case, if we consider e.g. *HML* with recursion or the μ -calculus, we may attempt to combine our work with existing results on characteristic formulas [2]. In that setting, instead of modal language properties, we could focus on compositionality of single formulas.

References

- [1] L. Aceto, W.J. Fokkink & C. Verhoef (2001): *Structural operational semantics*. In (J.A. Bergstra, A. Ponse and S.A. Smolka, eds) *Handbook of Process Algebra*, Elsevier, pp. 197–292.
- [2] L. Aceto, A. Ingolfsson & J. Sack (2009): *Characteristic Formulae for Fixed-Point Semantics: A General Framework*. In (D. Gorla and S. Fröschle, eds) *Proc. EXPRESS’09, EPTCS 8*, pp. 1–15.
- [3] P. Blackburn, M. de Rijke & Y. Venema (2001): *Modal Logic*. Cambridge University Press.
- [4] B. Bloom, W.J. Fokkink & R.J. van Glabbeek (2004): *Precongruence formats for decorated trace semantics*. *ACM Transactions on Computational Logic* 5(1), pp. 26–78.
- [5] W.J. Fokkink, R.J. van Glabbeek & P. de Wind (2006): *Compositionality of Hennessy-Milner logic by structural operational semantics*. *Theoretical Computer Science*, 354(3), pp. 421–440.

- [6] R.J. van Glabbeek (2001): *The linear time – branching time spectrum I; the semantics of concrete, sequential processes*. In J.A. Bergstra, A. Ponse & S.A. Smolka, editors: *Handbook of Process Algebra*, Elsevier, pp. 3–99.
- [7] M. Hennessy & R. Milner (1985): *Algebraic laws for non-determinism and concurrency*. *Journal of the ACM* 32(1), pp. 137–161.
- [8] C. Verhoef (1995): *A congruence theorem for structured operational semantics with predicates and negative premises*. *Nordic Journal of Computing* 2, pp. 274–302.